DOCUMENT RESUME

ED 125 554 IR 003 638

AUTHOR Stifle, J. E.
TITLE A Preliminary Report on the PLATO V Terminal.
INSTITUTION Illinois Univ., Urbana. Computer-Based Education Lab.
SPONS AGENCY Advanced Research Projects Agency (DOD), Washington, D.C.
REPORT NO CERL-R-X-46
PUB DATE Apr 76
CONTRACT USA-DAHC-15-73-C-0077
NOTE 54p.
AVAILABLE FROM PLATO Publications, Computer-Based Education Research Lab, 252 Engineering Research Lab, University of Illinois, Urbana, Illinois 61801 ($1.50)

EDRS PRICE MF-$0.83 HC-$3.50 Plus Postage.
DESCRIPTORS Autoinstructional Aids; *Computers; Computer Science; *Display Systems; *Electronic Equipment; *Input Output Devices; Instructional Media; Instructional Technology; Programing; *Technological Advancement; Time Sharing
IDENTIFIERS Computer Terminals; PLATO V; Programmed Logic for Automatic Teaching Operations

ABSTRACT

This report is a preliminary description of a prototype of a second generation version of the PLATO IV (Programmed Logic for Automated Teaching Operations) student terminal. Development of a new terminal has been pursued with two objectives: to generate a more economic version of the PLATO IV terminal, and to expand capacities and performance of student terminals. The terminal design is a miniature computer system containing all the standard features of large computer systems including a processing unit, memory, and an input-output structure. This miniature computer system performs as a PLATO terminal by attaching a plasma display panel to the terminal. The computer is programable and can execute programs locally. A library of programs and a set of self-diagnostic routines to be used to maintain the terminal are also being developed. (CH)

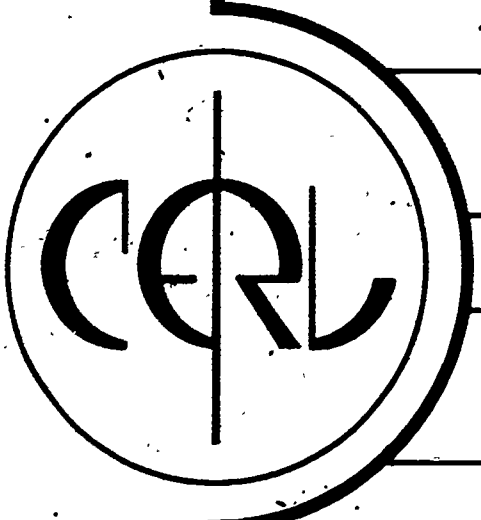
IR

CERL Report X-46

April 1976

A PRELIMINARY REPORT ON THE PLATO V TERMINAL

J. E. STIFLE



Computer-based Education Research Laboratory

University of Illinois Urbana Illinois

This manuscript was prepared with partial
support from the Advanced Research Projects
Agency (US Army/DAHC 15-73-C-0077) and from
the State of Illinois.

ACKNOWLEDGMENTS

The success of this program is due in large part to the efforts of several people. Included in this group are Leonard Hedges, who performed the mechanical layout and design and supervised the construction, and Donald Hartman, who did the actual construction. Michael Hightower helped write the terminal resident and also designed the ROM programming system. Roger Johnson and his group were responsible for the development of the parallel version of the plasma panel.

A special thanks is due Donald Lee, who wrote a PLATO 8080 assembler which greatly simplified the task of developing the terminal resident program.

Thanks also to Sheila Knisley, who did all of the typing, Roy Lipschutz, who did the drafting, and to Bruce Sherwood and Paul Tenczar for their helpful comments.

1. Introduction

1.0 Program Description

This report is a preliminary description of a prototype of a second generation version of the PLATO IV student terminal.. This prototype has evolved from a program which has been pursued for the last two years and which has two primary objectives: 1) the development of a more economical version of the existing PLATO IV terminal and 2) the design of a second generation (PLATO V) terminal with greatly expanded capabilities and improved performance. The development program has, from the beginning, concentrated on developing a terminal architecture which can simultaneously satisfy both of the above stated objectives.

A terminal architecture has been designed which is actually a miniature computer system containing all of the standard features of large computer systems including a processing unit, memory and an input-output (IO) structure. This miniature computer system is made to perform as a PLATO terminal by simply attaching a plasma display panel to the IO structure.

As with any computer system, the contents of the memory define the operating characteristics of the terminal. It is this programmable characteristic feature that enables the same terminal architecture to behave as a PLATO IV terminal or as a more powerful terminal with additional capabilities. (It is even possible to operate this terminal as a standard ASCII unit.)

The memory hierarchy in the prototype presently contains 4K words (8 bits) of read-only memory (ROM) and 8K words of random access memory (RAM) and is expandable to a total of 64K words. The RAM portion of the memory can be used to store data to be displayed or programs to be executed locally (at the terminal). This concept of locally-executable programs represents a marked departure from the existing PLATO IV terminal where the memory can only be used to store data.

It should be emphasized that the terminal described here is a prototype and is still undergoing changes as the program develops; therefore, from time to time this report will refer to changes that are being considered but which are not yet implemented.

In addition to the design of the terminal itself, several closely related tasks are being pursued. These include the development of a library of programs which can be loaded into the terminal, the investigation of the system implications of using such a terminal, particularly on the PLATO formatting program, and a study of the possibility of transferring some processing tasks from the central system to the terminal. Additional efforts are being directed at developing a set of self-diagnostic routines which can be used to maintain the terminal.

The prototype terminal is shown in Figure 1.

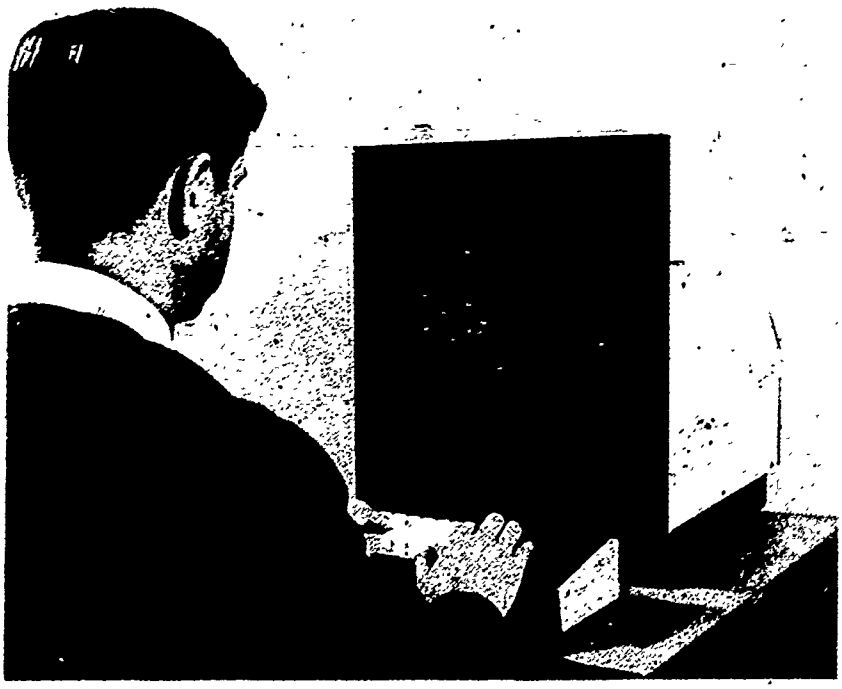


Figure 1. Prototype Terminal

2. Terminal Description

2.0 Terminal Processor Unit (TPU)

A block diagram depicting the main functional units of the terminal is shown in Figure 2.0. Operation of the terminal is under the supervision and control of the Terminal Processing Unit (TPU) which is an Intel 8080 microprocessor constructed on a single LSI chip. This chip is an 8-bit parallel processor with a basic 2μs instruction cycle time and contains six 8-bit data registers, an 8-bit accumulator, four flag bits, a 16-bit program counter and a 16-bit stack pointer register which is used to control a last in-first out stack contained in memory.

An 8-bit bidirectional data bus (D) connects the TPU to the other functional units within the terminal and to I/O devices external to the terminal. The data bus acts as a highway over which information flows between internal (and external) portions of the terminal. In addition to performing display generation calculations, the TPU must manage the flow of information on the highway. Also present but not shown is a 16-bit address bus (A) which is used to specify memory and input-output device addresses.

The 5-bit status register specifies the type of machine cycle presently being executed by the TPU. The five types of machine cycles are input, output, memory read, memory write, and interrupt. The outputs of this register are used to assist in controlling the attachment of units to the D bus.

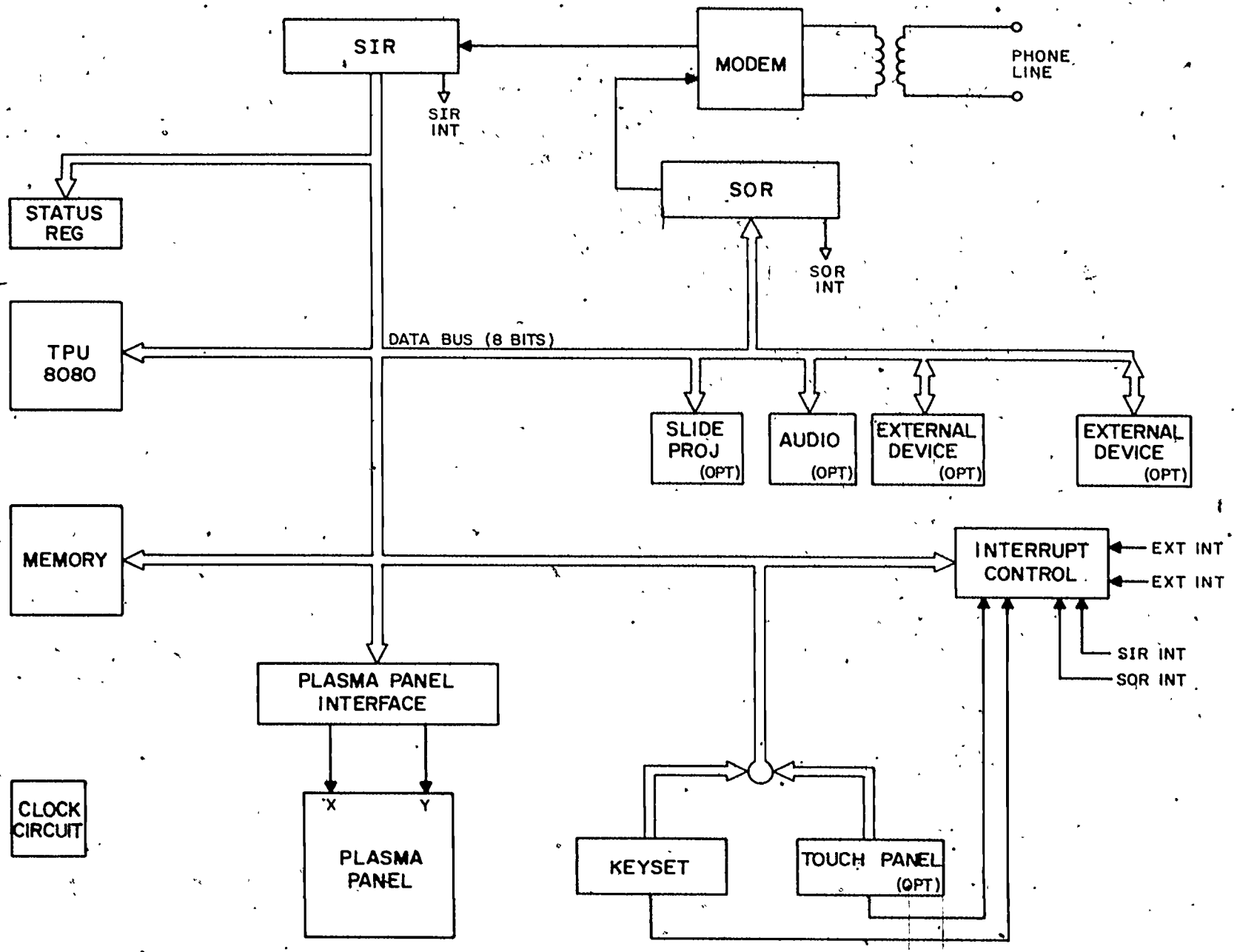


Figure 2.0. Terminal Architecture

2.1 Serial Input Register (SIR)

Data from the central computer arrives as 21-bit words at a rate of 60 words per second. Each word contains a message start bit and 20 information bits. The modem removes the message start bit and passes the information bits to the SIR where they are accumulated. After a 20-bit word is assembled, the SIR interrupt is generated indicating to the TPU the presence of a new word in the SIR. The TPU responds to the interrupt by transferring the contents of the SIR into the TPU. The data in the SIR is disassembled into 3 eight-bit bytes for transfer to the TPU. The format of these bytes along with all other word formats will be discussed later.

The SIR automatically performs a parity check on each word as it is being accumulated. If the word contains an error, the Parity Error/Lost Data flag is set while if there is no error this flag is cleared. This flag is transferred to the TPU as part of the SIR data. In the event the TPU fails to respond to the SIR interrupt prior to the arrival of another word, the Parity Error/Lost Data flag is set, the missed word is discarded and the new word is assembled in the SIR.

2.2 Serial Output Register (SOR)

The SOR is a 12-bit shift register used to transmit data from the terminal to the central computer. This register is loaded by the TPU in two parts as shown in Figure 2.1. Following the loading of the lower 8 bits, the entire contents of the SOR are transmitted to the central computer. The upper bits of the SOR must, therefore, always be loaded prior to the lower 8 bits. Loading the lower 8 bits of the SOR will set the SOR Ready flag to "zero," indicating the register is busy. This flag will remain "zero" until the contents of the SOR have been transmitted, at which time it is automatically set to "one." Each time this flag is set to "one" the SOR interrupt is generated, which informs the TPU that the SOR is available for use. This flag is also transferred to the TPU as a part of the SIR data.

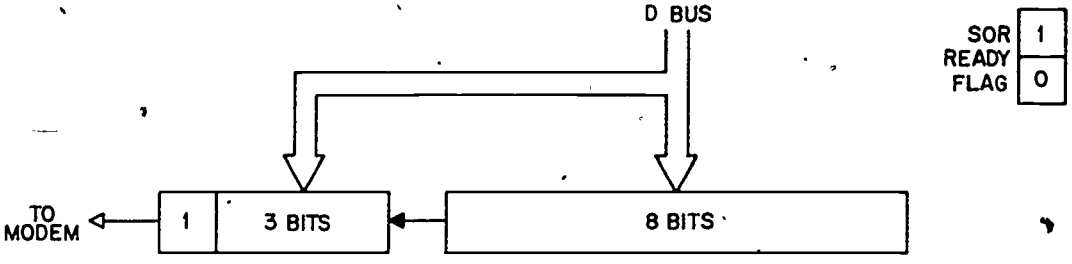


Figure 2.1. SOR Word Assembly

2.3 Interrupt Control

All service requests to the TPU are made via the Interrupt Control Unit (ICU). A device requesting service originates an interrupt and presents it to the ICU. Within the ICU the interrupt requests have a wired-in priority, and the ICU will pass to the TPU the interrupt request having the highest priority.

When an interrupt request is accepted by the TPU, normal program sequencing is halted; the present contents of the Program Counter (PC) are pushed into the stack in memory; and a RST (unconditional jump) instruction to location 70 is forced into the TPU instruction register. Following the interrupt, the TPU inputs a word from the ICU which contains the address of the interrupting source.

The ICU presently provides for eight interrupt sources with the priority and addresses shown in Table 1.

Request		RST Address (octal)
SIR	highest priority	000
KST		001
TP		002
SOR		003
EXT0		004
EXT 1-3		005-007

Table 1. Interrupt Priority

The processing time for the various types of SIR words is given in the discussion of word formats in section 3. Processing time for KST, TP, or EXT interrupts is 175μs while the SOR requires 120μs.

Within the TPU is an interrupt enable flag which must be set before any interrupt requests will be accepted from the ICU. This flag can be set by the EI (enable interrupt) and reset by the DI (disable interrupt) instructions. Each time an interrupt is accepted by the TPU, this flag is automatically reset thus disabling further interrupts until set by an EI instruction. This flag is also reset by the CLEAR switch.

The ICU contains an 8-bit Interrupt Mask Register, each bit of which is associated with an interrupt source as shown in Figure 2.2. An interrupt is enabled if the associated bit in the Mask Register is a "one," disabled if it is a "zero." An interrupt will not be passed through the ICU unless it is enabled. The TPU can, therefore, selectively enable or disable interrupts by the data loaded into the Mask Register. The data to be loaded into the Mask Register is maintained in a protected location in memory. An interrupt is said to be 'armed' if the associated bit in this location is a "one," 'disarmed' if it is a "zero." The resident program can selectively set (arm) or clear (disarm) bits in this word before sending it to the Mask Register. Thus, an interrupt may be armed but temporarily disabled by the resident.

Figures 2.3 and 2.4 contain the flow charts describing the processing for each type of interrupt. In these diagrams the symbol RTN means return to the program in progress when the interrupt occurred; SAVE means save the present contents of the registers and flags in the stack. In performing RTN the saved information will be restored to the register, and interrupts will be enabled as the are then armed.

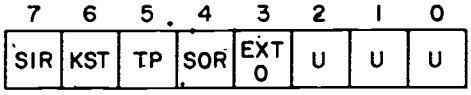


Figure 2.2. Interrupt Mask Register Bit Assignment

2.4 Memory

The terminal memory presently consists of 12,288 words, 4096 of which are ROM and 8192 of which are RAM. The lower 2K words of ROM contain the terminal resident program. (The present version of the resident occupies approximately 2000 words.)

The resident contains all of the programs to emulate the existing PLATO IV terminal plus programs for processing interrupts, handling IO, error processing, and some additional character mode features. These additional character features include the ability to write characters from right to left and vertically in both directions. Also included is the ability to plot characters that are twice the size of the standard character set. New programs are presently being developed which may be incorporated in the resident to become standard features of the terminal.

The upper 2K words of ROM contain the data for the ROM characters existing in the present PLATO IV terminal.

The RAM memory begins at location 4096 and extends through location 12,287. The first 2048 words of RAM, location 4096 through 6143, are usually reserved for the loadable character set data, M2 and M3, as in the existing PLATO terminal. (Sixteen 8-bit words are required per character.)

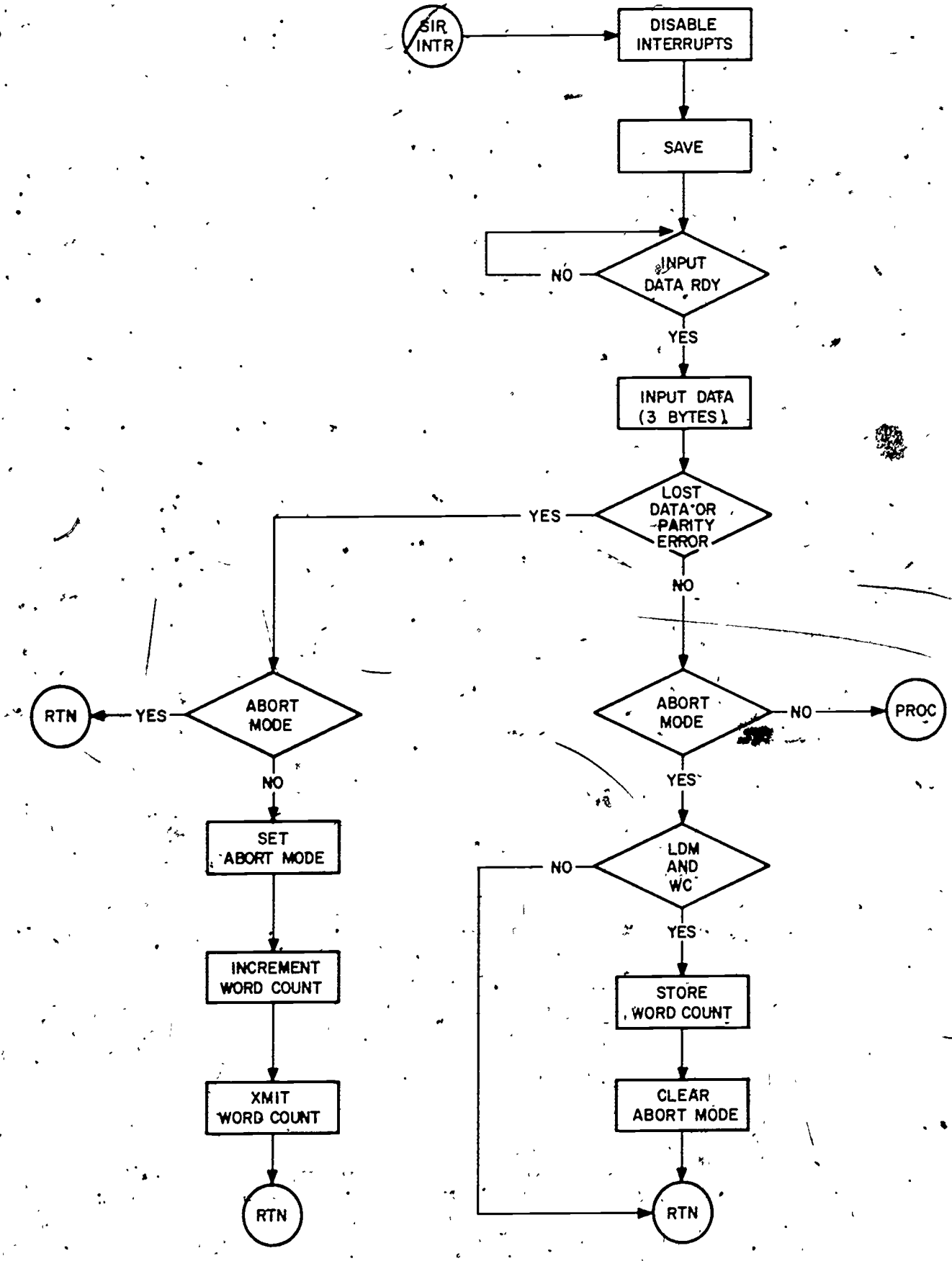


Figure 2.3 SIR Interrupt

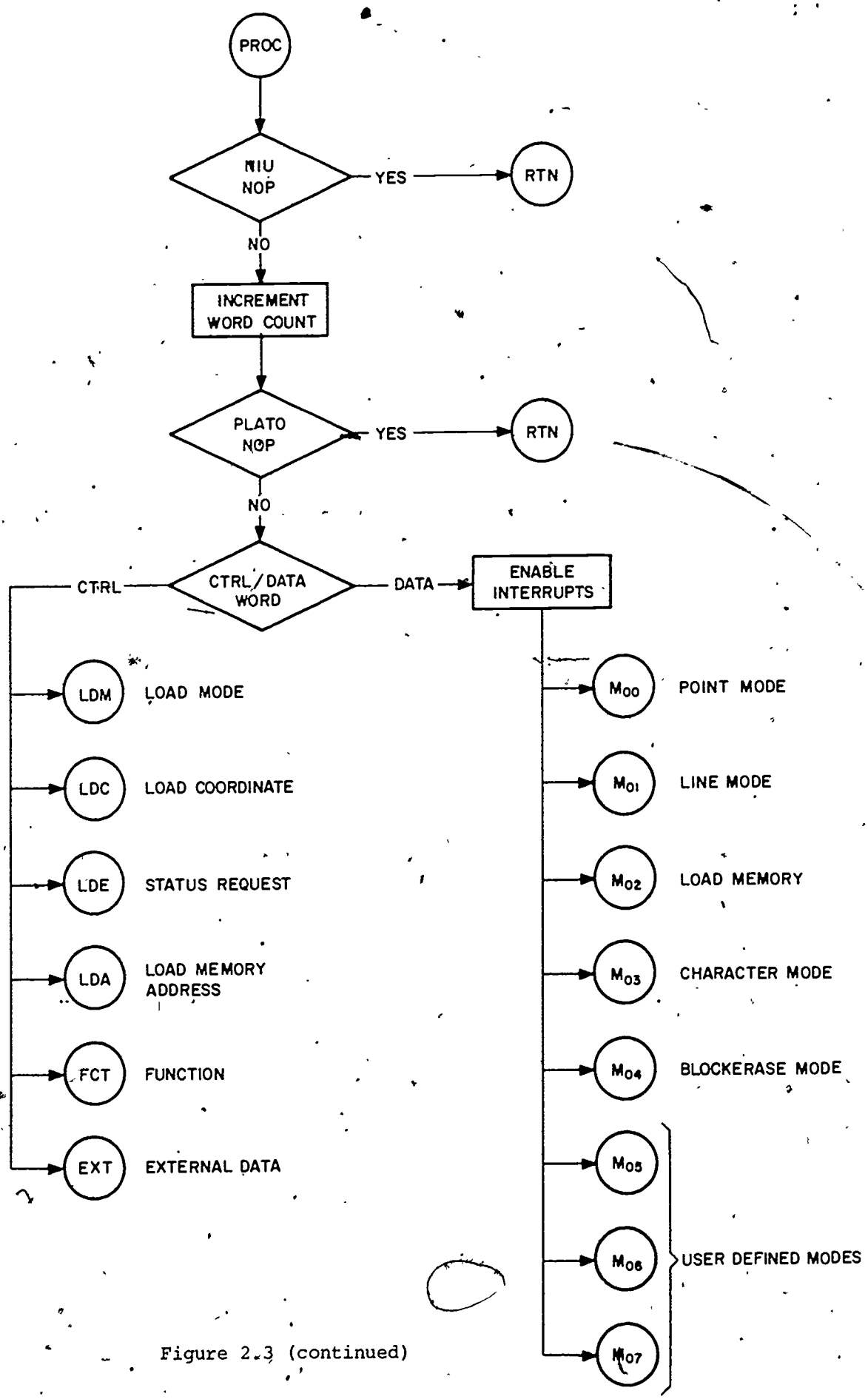


Figure 2.3 (continued)

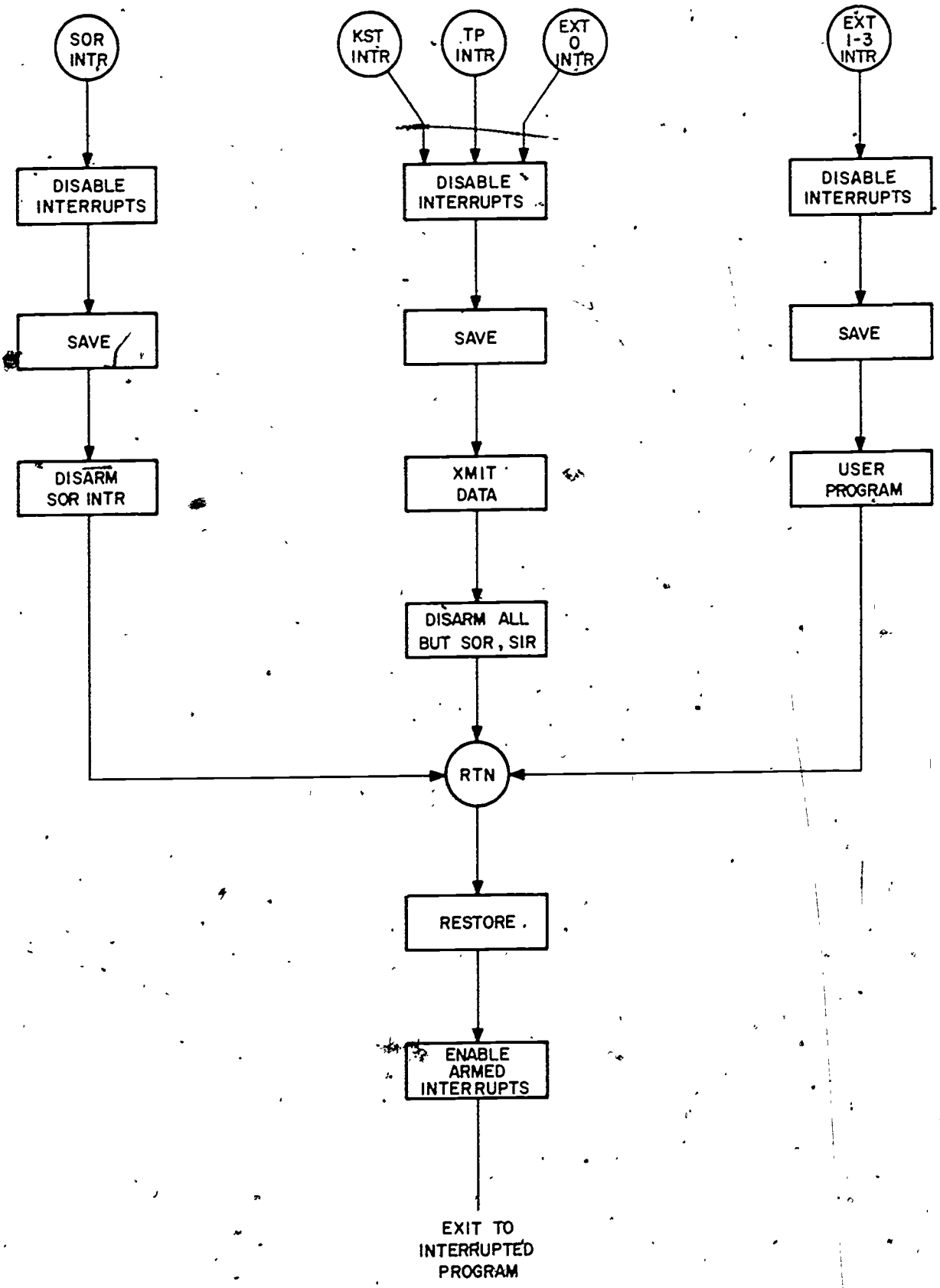


Figure 2.4 KST, TP, SOR and EXT Interrupts

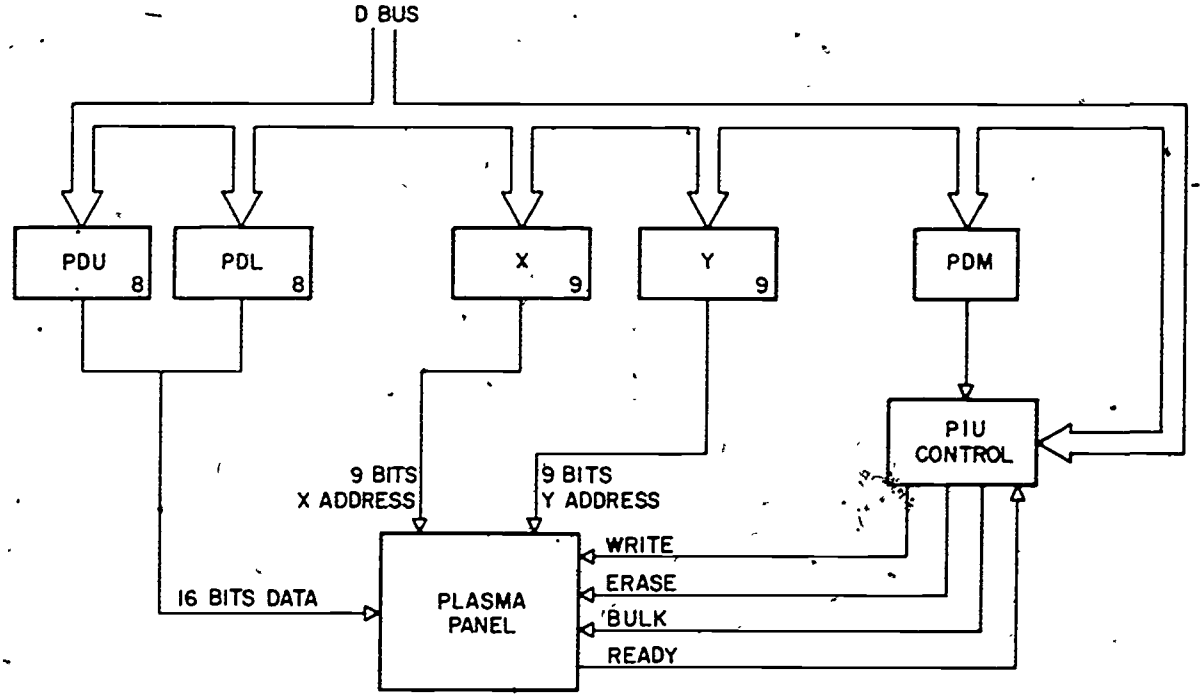


Figure 2.5. PIU

The top 64 words of RAM are reserved for use by the resident program. Just below this reserved section is the portion of memory used as a last in-first-out stack by the TPU. The terminal resident program will prevent the loading of data from the central computer into any area of the memory currently occupied by the stack.

2.5 Panel Interface Unit (PIU)

The PIU contains the registers and control circuits required to efficiently attach a plasma panel to the D bus. The PIU, shown in Figure 2.5, contains the 9-bit x and y panel address registers, a 16-bit parallel data register (PDL/U), a 3-bit panel display mode (PDM) register and the write-erase control circuits. Data for the registers and control information is supplied to the PIU via the D bus. The x and y registers are bidirectional counters which can be independently controlled by the TPU. The parallel data register consists of the 8-bit parallel data upper (PDU) and parallel data lower (PDL) registers. The PDU/L registers are used only when operating parallel input plasma panels.

The format of the PDM register is shown in Figure 2.6. Bits 0 and 1 specify the write/erase mode, and bit 2, the panel operating mode. If bit 2 is "zero," the panel is operating in the serial mode, and the contents of the x and y register specify the address of a point to be written or erased. If bit 2 is "one," the panel is operating in the parallel mode, and the contents of PDL/U will be written or erased on the panel at the address specified by the contents of the x and y registers. The data will be displayed in a vertical column with bit 0 of PDL at the bottom and bit 7 of PDU at the top.

Information is written on the display if $WE_1 = 1$ or erased if $WE_0 = 0$. The use of the WE_1 bit is explained in Section 3.5 in the discussion of Mode 3.

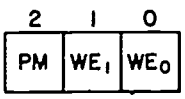


Figure 2.6. PDM Register

2.6 Input-Output

The present prototype terminal provides for the attachment to the D bus of up to 32 input and 32 output devices. Input devices include a keyset (standard), touch panel (optional) and other optional input equipment. Output devices, all optional, include a random-access slide projector for the projection of slide images on the rear of the plasma panel, a random-access audio unit which can play back to the terminal user pre-recorded audio messages, a ROM programmer, and a floppy disk system. Other user-defined output devices may also be attached. Data rates in excess of 25K bytes per second may easily be accomplished on the IO Bus.

3. Operating Modes

3.0 PLATO Word Format

The data to be processed by the terminal consists of 20-bit words (with start bit removed) with the format shown in Figure 3.0.

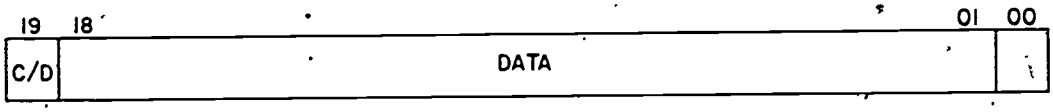


Figure 3.0. Terminal Word Format

Bit 00	Parity bit - even parity
Bits 01 - 18	Data
Bit 19	Control bit - 0 = control word - 1 = data word

Terminal words may be of two types: control words and/or data words. Data words (c = 1) contain the data to be processed by the terminal while control words (c = 0) are instructions used to establish operating conditions within the terminal.

The SIR checks each word for parity and disassembles the word into three 8-bit bytes for transfer into the TPU. The format of these bytes along with the TPU input address assignment is shown in Figure 3.1.

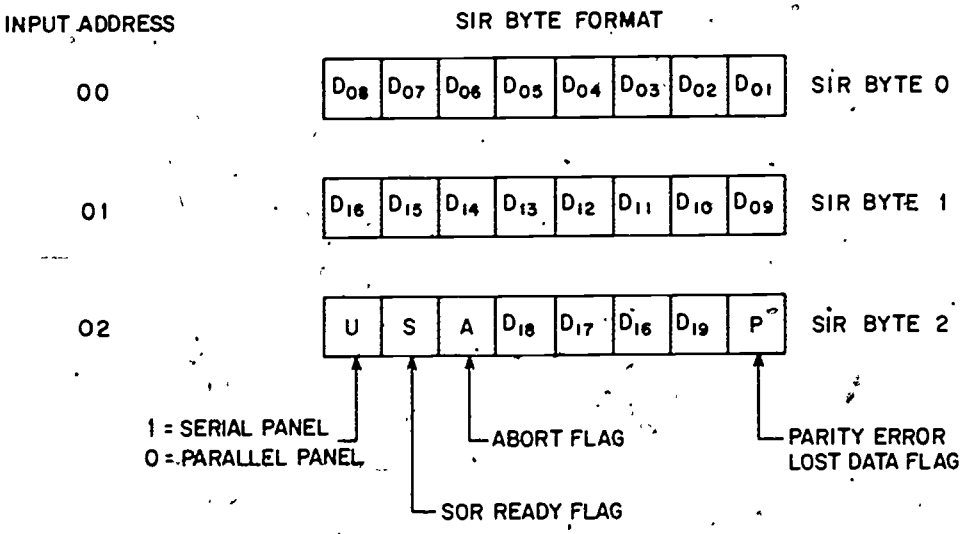


Figure 3.1. SIR Byte Format

In addition to SIR data, byte 2 contains three IO status flags. The Parity Error/Lost Data flag indicates that either the present SIR word contains an error or that the TPU failed to input one or more previously received SIR words.

The ABORT Mode flag indicates the error mode status of the terminal. If this flag is "zero," the terminal is operating normally, while if it is "one," the terminal is in the ABORT mode of operation. The TPU maintains a record (word count) of the number of non-NOP words received. Each time a non-NOP word is transferred into the TPU, the word count is incremented by 1. Upon receipt of a word containing a parity error or an indication of lost data, the TPU automatically transmits the value of the word count to the computer center, sets the ABORT flag and enters the ABORT mode of operation. The value of the word count transmitted will indicate to the center the address of the word containing the error or the word that was lost.

Once in the ABORT mode, the terminal will refuse to accept any further information except for a LDM instruction (described later). Receipt of an LDM instruction with bit 14 a "one" will clear the ABORT flag and return the terminal to normal operating mode. This method of error control prevents the terminal from processing data in the wrong mode in the event an erroneous mode change word is received.

The SOR Ready flag indicates the present status of the SOR. If this bit is "zero," the SOR is busy transmitting data, while if it is "one," the SOR is available for use. Before loading data into the SOR, the TPU examines this flag to determine SOR availability.

In the discussion of word formats which follows, the processing times quoted include the time required to interrupt and save the present status of the TPU, the execution time for the word and the time required to restore the TPU to its original state.

The quoted times assume a TPU clock frequency of 2 MHz (.5μs per state) and a maximum memory access time of .5μs.

3.1 Control Word Formats

The PLATO control word format is shown in Figure 3.2.

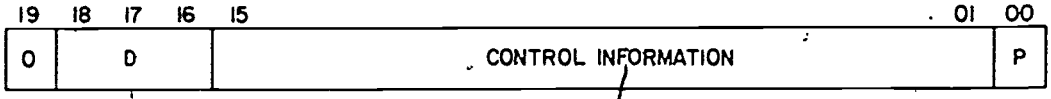
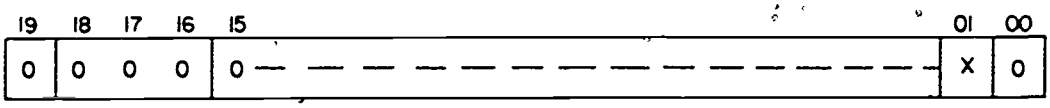


Figure 3.2. Control Word Format

Bits 01 - 15 Control Information

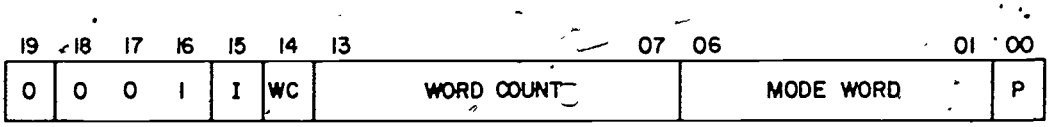
Bits 16 - 18 Type of Control Word

D = 000 (NOP)



This word is a NOP (no-operation) instruction. There are two types of NOP words, those generated by the NIU and those generated by PLATO software. The PLATO NOP will cause the terminal word count to be incremented while the NIU NOP will not affect the terminal status in any way.

D = 001 (LDM) Load Mode (267μs)



This instruction establishes the operating mode of the terminal. For each mode of terminal operation there is an associated mode word (bits 01 - 06) which directs the processing of incoming data. Once placed in a given mode, the terminal remains in that mode until receipt of a new LDM instruction.

If bit 14 (WC) of the LDM word is "one," the word count register will be set to the value specified by bits 07 - 13. It is the receipt of this instruction with bit 14 set which will restore the terminal to normal mode if it is in the ABORT mode. This is the only instruction which the TPU will accept if it is in ABORT mode. Receipt of the LDM instruction while in the ABORT mode will clear the ABORT flag and initialize the word count but will not alter the terminal processing mode.

Bit 15 of the LDM word is used to actuate or inhibit external devices attached to the terminal. Receipt of an LDM word with bit 15 a "one" will disable the interrupts from all external devices except the keyset. They will remain inhibited until receipt of an LDM word with bit 15 a "zero."

The terminal mode word format is shown in Figure 3.3.

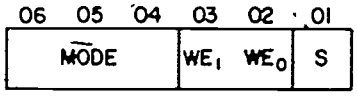


Figure 3.3. Mode Word Format

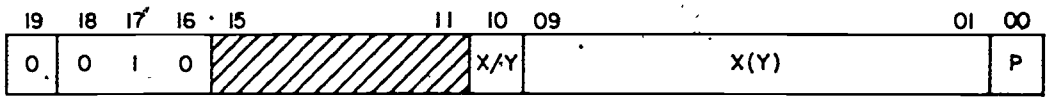
Bit 01 — Screen Command. If this bit is "1," the entire display panel is erased.

Bits 02 - 03 — Select write or erase function in the PIU as follows:

WE_1	WE_0	
0	0	Erase, write character background
0	1	Write, erase character background
1	0	Erase, suppress character background write
1	1	Write, suppress character background erase

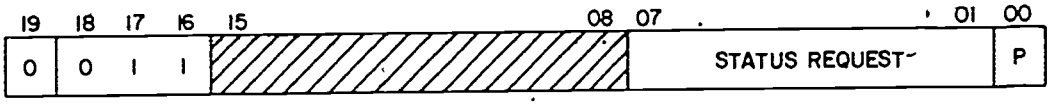
Bits 04 - 06 — Specify terminal processing mode, The modes are described later.

D = 010 (LDC) Load Co-ordinate (225μs)



This instruction loads the X register (bit 10 = 0) or the Y register (bit 10 = 1) with bits 01 - 09. Bits 11 - 15 are unused. It is anticipated that this instruction will also be used for relative addressing. Bits will be inserted in the unused portion of the word to indicate relative or absolute addressing. Used in relative addressing, bits 10 - 09 would be added to or subtracted from the present contents of the specified register.

D = 100 (LDE) Status Request (267μs)

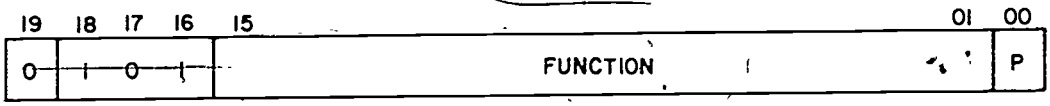


It is proposed that this word, presently used as an ECHO code, be used to request the status of various conditions within the terminal. The terminal would respond to the status request command by issuing a unique status response code. (See section 3.7.) In the present PLATO system the terminal merely echoes the data contained in bits 01 - 07.

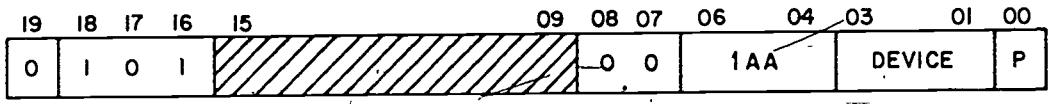
D = 100 (LDA) Load Memory Address (229μs)

This instruction loads the Memory Address Register (MAR). This data word specifies the first storage address to be used upon entry into a Mode 2 operation. RAM memory begins at location 4096 (10000_8) which is the lowest useable address. The terminal resident will prevent the loading of data into locations above 27700_8 or into any locations above the present value of the stack pointer register.

D = 101 (FCT) Function (223μs)



This instruction is proposed for use to establish special functions within the terminal. One use would be to specify an external output device to which all subsequently received EXT words would be sent. The format of the FCT instruction in this case is shown below.

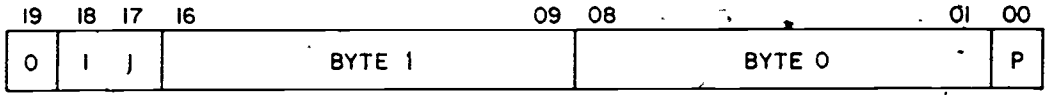


Bits	
01-03	specify external device address
04-06	specify external channel number

AA	External channel
00	4
01	5
10	6
11	7

Channels 0-3 are used internal to the terminal and are not available for use by this instruction.

D = 11x (EXT) Load External (275μs)



This instruction transfers two 8-bit bytes, byte 0 first, to the external device selected by a previous FCT instruction.

In the present PLATO IV terminal only 15 bits can be sent to an external device (see Appendix).

3.2 Processing Modes - Mode 0

In normal operation, the terminal is assigned an operating mode by sending it a LDM instruction followed by all of the data to be processed in that mode.

The terminal resident program contains the programs for processing data in the four modes presently used in PLATO IV. In addition, up to four additional user-defined-mode programs can be loaded into RAM.

Mode 0 is a point plotting mode. Each mode 0 data word, Figure 3.4, specifies the address of a point on the panel to be written or erased. The W/E_0 bit in the mode word determines which operation is performed.

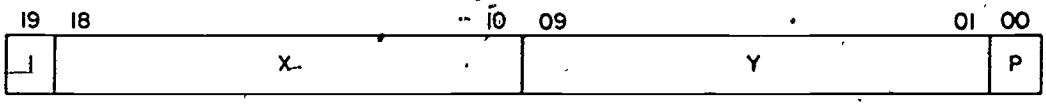


Figure 3.4. Mode 0 Data Word

The processing time for a Mode 0 word is 238μs.

3.3 Mode 1

Mode 1 is a line drawing mode. Each data word, Figure 3.5, specifies the terminal coordinates of a line, the origin of which is contained in the X and Y registers.

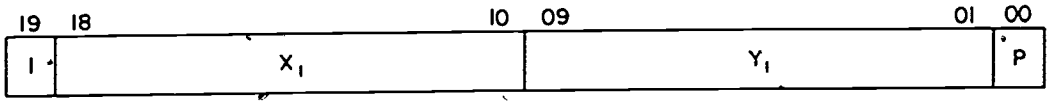


Figure 3.5. Mode 1 Data Format

The terminal point of a given line is also interpreted as the origin of the next line. Line origins may be relocated, however, by the use of the LDC command without exiting from Mode 01.

The processing time for a Mode 1 word ranges from 1ms for a line length of one dot to 11.1ms for the maximum line length of 512 dots.

3.4 Mode 2

Mode 2 is a load memory mode. Each Mode 2 data word, Figure 3.6, contains two eight-bit words to be stored in RAM memory. These words are stored, lower first, in two successive locations starting with the present contents of the memory address register (MAR). After each 8-bit word is stored, the MAR is automatically incremented by 1. The resident program will refuse to load data into memory above the present value of the stack pointer register or above location 27700_8. An attempt to load into those areas will cause the store operation to be discarded and an illegal store code will be sent to the computer center. A longitudinal parity check will be added to the Mode 2 resident which will permit the TPU to call for the retransmission of any block of data containing an error.

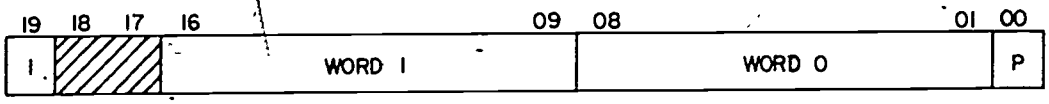


Figure 3.6. Mode 2 Data Word

The stored data when displayed on the panel appears as a vertical column with bit 01 of word 0 at the bottom and 16 of word 1 at the top.

The processing time for a Mode 2 word is 288μs.

3.5 Mode 3

Mode 3 is a character plotting mode. The data words in this mode contain three 6-bit character codes as shown in Figure 3.7. Each code selects one of 63 characters from one of the character sets contained in the terminal.

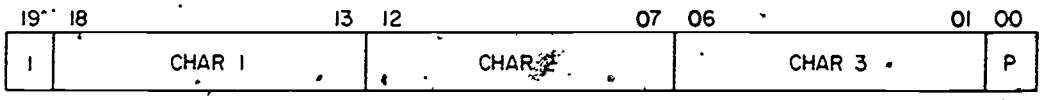


Figure 3.7. Mode 3 Data Word

The terminal presently provides for up to eight character sets of 63 characters each. Character sets M0 and M1 are contained within the terminal resident and hold the characters shown in Table 2. The other character sets contain user-defined characters and are stored in RAM beginning at location 10000_8.

The contents of the character memories are processed in Mode 3 as 63 arrays of sixteen 8-bit words. The contents of 16 consecutive addresses are displayed as one character within a matrix as shown in Figure 3.8. The top three rows and bottom row of all character matrices from M0 and M1 are always unfilled.

ADDRESS (OCTAL)	M0 CHAR	M1 CHAR	ADDRESS (OCTAL)	M0 CHAR	M1 CHAR
0	:	#	40	5	↑
1	a	A	41	6	→
2	b	B	42	7	↓
3	c	C	43	8	←
4	d	D	44	9	~
5	e	E	45	+	Σ
6	f	F	46	-	Δ
7	g	G	47	*	∪
10	h	H	50	/	∩
11	i	I	51	(	{
12	j	J	52	)	}
13	k	K	53	$	&
14	l	L	54	=	≠
15	m	M	55	SP	SP
16	n	N	56	,	|
17	o	O	57	.	°
20	p	P	60	÷	≡
21	q	Q	61	[	α
22	r	R	62	]	β
23	s	S	63	%	δ
24	t	T	64	×	λ
25	u	U	65	=	μ
26	v	V	66	'	π
27	w	W	67	"	ρ
30	x	X	70	!	σ
31	y	Y	71	;	ω
32	z	Z	72	<	≤
33	0	~	73	>	≥
34	1	¨	74	_	θ
35	2	^	75	?	@
36	3	´	76	»	\
37	4	`	77	UNCOVER	UNCOVER

Table 2. ROM Characters

The contents of any character memory can be enlarged via selection of character size 1 (size 0 is normal size). Selection of size 1 will result in a 2X magnification of the characters. Figure 3.9 illustrates characters drawn in size 1. All character format operation will be automatically adjusted when using size 1 characters.

Character write/erase is specified by the write/erase bits WE0, WE1 in the mode word. (See LDM instruction.) If WE0 = 1, characters are written; if WE0 = 0, characters are erased. The inverse of the operation called for by WE0 will be performed on the background or unfilled portion of the character matrix if WE1 = 0, while if WE1 = 1, the background remains unaltered.

Character plotting speed ranges from a minimum of 316 characters per second (3.16ms/char) using a serial plasma panel up to 2020 characters per second (495μs/char) using a parallel plasma panel.

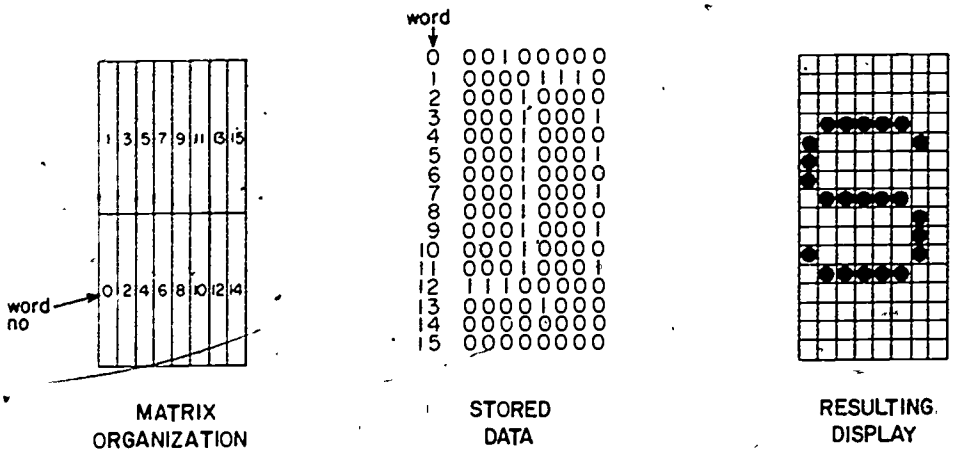


Figure 3.8. Character Matrix

This terminal is a prototype for a PLATO V terminal. It is operated under the supervision and control of an INTEL 8080 microprocessor. It is actually a miniature interrupt driven time-shared computer system with a plasma panel attached to the i/o bus. The terminal contains 12k bytes of memory, 8k of which are RAM which can be used to store data or programs which can be executed at the terminal.
ABCDEFGHIJKLMNOPQRSTUVWXYZ ⩾#?!
0123456789<> [] $%_'* () +-÷×⇐,./;π

Figure 3.9. "Boldface" Character Font

There are several non-plotting control characters available for formatting the display of data in Mode 3. These control characters may be accessed via the use of the "uncover" code (77). Upon receipt of a 77 code, the terminal interprets the next character code as a control character instead of a character to be plotted. Following execution of the control character, normal plotting mode is resumed. If several uncover codes are sent in sequence, the first non-uncover code will be treated as the control character.

The operations performed by each of the control characters are shown in Table 3. In the case of some characters, the operation performed is a function of the character memory, the plotting mode, horizontal or vertical, and the plotting direction, forward or reverse, which is being used.

Horizontal plotting mode is set by the 30 code; vertical plotting, by the 31 code. Forward plotting direction, set by the 32 code, is from left to right in horizontal mode and from bottom to top in vertical mode. Reverse plotting direction set by the 33 code, is from right to left in horizontal mode and from top to bottom in vertical mode.

The CRMØ and CRM1 (70, 71) each perform a carriage return and then set the panel address equal to the contents of memory location MARGØ or MARG1, respectively. The TABØ and TAB1 (72, 73) codes are dynamically adjustable tab settings. The contents of MARGØ,1 and TABØ,1 are specified by the user.

OCTAL CODE	FUNCTION	SIZE Ø				SIZE 1			
		HORIZONTAL		VERTICAL		HORIZONTAL		VERTICAL	
		FWD	RVS	FWD	RVS	FWD	RVS	FWD	RVS
10	Backspace	x-8	x+8	y-8	y+8	x-16	x+16	y-16	y+16
11	Tab	x+8	x-8	y+8	y-8	x+16	x-16	y+16	y-16
12	Line Feed	y-16	y-16	x+16	x+16	y-32	y-32	x+32	x+32
13	Vertical Tab	y+16	y+16	x-16	x-16	y+32	y+32	x-32	x-32
14	Form Feed	x←Ø y←496	x←5Ø4 y←496	y←Ø x←15	y←5Ø4 x←15	x←Ø y←480	x←496 y←480	y←Ø x←31	y←496 x←31
15	Carriage Return	x←Ø y-16	x←5Ø4 y-16	y←Ø x+16	y←5Ø4 x+16	x←Ø y-32	x←496 y-32	y←Ø x+32	y←496 x+32
16	Superscript	y+5	y+5	x-5	x-5	y+10	y+10	x-10	x-10
17	Subscript	y-5	y-5	x+5	x+5	y-10	y-10	x+10	x+10
20	Select MØ	select character memory Ø (ROM)							
21	Select M1	select character memory 1 (ROM)							
22	Select M2	select character memory 2 (RAM)							
23	Select M3	select character memory 3 (RAM)							
24	Select M4	select character memory 4 (RAM)							
25	Select M5	select character memory 5 (RAM)							
26	Select M6	select character memory 6 (RAM)							
27	Select M7	select character memory 7 (RAM)							
30	Horizontal	select horizontal plot mode							
31	Vertical	select vertical plot mode							
32	Forward	select forward plot direction							
33	Reverse	select reverse plot direction							
34	Select Size Ø	select normal-size characters							
35	Select Size 2	select large-size characters.							
70	CRMØ	y-16 x←MARGØ		x+16 y←MARGØ		y-16 x←MARGØ		x+16 y←MARGØ	
71	CRM1	y-16 x←MARG1		x+16 y←MARG1		y-16 x←MARG1		x+16 y←MARG1	
72	TABØ	x←TABØ		y←TABØ		x←TABØ		y←TABØ	
73	TAB1	x←TAB1		y←TAB1		x←TAB1		y←TAB1	
77	Uncover	next character is control character							

Table 3. Control Characters

3.6 Mode 4

Mode 04 is presently being considered for use as a block erase mode. In this mode each pair of data words would specify the corners of an area to be erased. The area erased is that enclosed by $|\Delta x| = |x_2 - x_1|$ and $|\Delta y| = |y_2 - y_1|$: If $x_2 = x_1$ and $y_2 - y_1$, a single point is erased while, if $x_2 = x_1$, a vertical line is erased, and if $y_2 = y_1$, a horizontal line is erased.

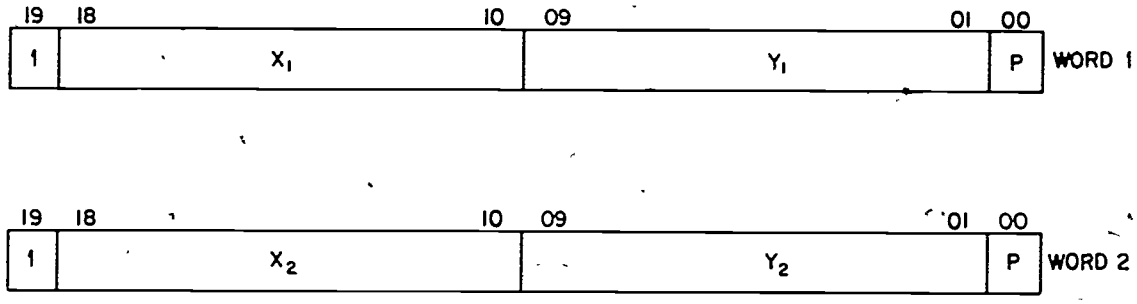


Figure 4.0. Mode 4 Word Format

3.7 SOR Word Format

Data transmitted from the terminal to the computer center consists of 12-bit words with the format shown in Figure 4.1.

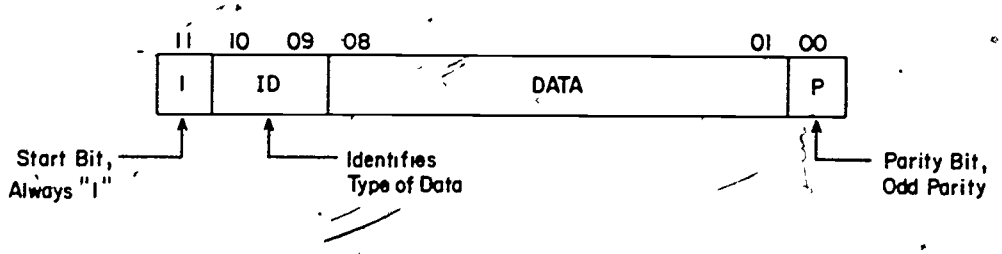


Figure 4.1. SOR Word Format

The format for the various types of data are shown in Figure 4.2. The Status Response word was formerly called ECHO code. It is proposed that the ECHO command would become a Status Request command and the present ECHO code would become the response to the Status Request. One of the Status Request codes would be called ECHO in which case the ECHO code would return exactly as in the present system.

Status Request code 160_8 is used to request terminal type. A Status Response code of 160_8 indicates an existing PLATO IV terminal while a response of 162_8 indicates an 8080-type terminal.

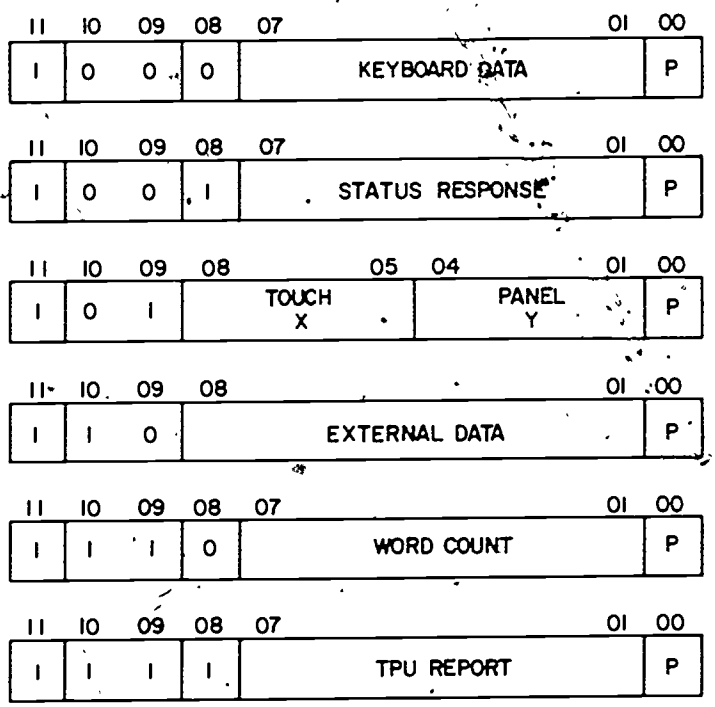


Figure 4.2. SOR Word Formats

The TPU report codes are used to inform the central computer of the occurrence of some special event within the terminal. A list of the presently used TPU report codes is shown in Table 4.

TPU Report (Octal)	Event Reported
000	Terminal has just been turned on
001	Not used
002	Reset (clear switch has been depressed)
003	Automatically load the terminal RAM with with some program (bootstrap)
004	An attempt to write in an illegal area or RAM during Mode 2
005	Longitudinal parity error occurred in Mode 2
006	Reject, terminal busy

Table 4. TPU Report Codes

4. Instruction Set[1]

Terminal processor instructions may be from one to three 8-bit bytes (words) in length with the bytes for a multiple byte instruction stored in successive memory location. The first byte always contains the instruction operation code while the second and third bytes contain operands or memory addresses.

The following symbols will be used in the detailed description of the instruction set.

Symbol	Definition
<B2>	Second byte of the instruction
<B3>	Third byte of the instruction
r	Scratch pad or memory register designation

r	Register
000	B
001	C
010	D
011	E
100	H
101	L
110	Memory
111	A (accumulator)

M	Memory location indicated by the contents of registers H and L
( )	Contents of location or register
.	Logical product
⊕	Exclusive "or"
V	Inclusive "or"
A_i	Bit i of the A-register
SP	Stack Pointer
PC	Program Counter
←	Is transferred to

[1] The instruction set described here is from the "Intel 8080 Microcomputer System Manual," January 1975, published by the Intel Corp., 3065 Bowers Ave., Santa Clara, CA 95051. The input-output device assignments are unique to this terminal.

Mnemonic	Op Code (octal)	Execution Time (μs)	Description of Operation
MOV r_1, r_2	$1r_1r_2$	2.5	$(r_1) \leftarrow (r_2)$ Load register r_1 with the content of r_2. The content of r_2 remains unchanged.
MOV M, r	16r	3.5	(M)←(r) Load the memory location addressed by the contents of registers H and L with the content of register r.
MOV r, M	1r6	3.5	(r)←(M) Load register r with the content of the memory location addressed by the contents of registers H and L.
HLT	166	3.5	On receipt of the Halt Instruction, the activity of the processor is immediately suspended in the STOPPED state. The content of all registers and memory is unchanged and the PC has been updated.
MVI r <B_2>	0r6	3.5	(r)←<B_2> Load byte two of the instruction into register r.
MVI M <B_2>	066	5.0	(M)←<B_2> Load byte two of the instruction into the memory location addressed by the contents of registers H and L.
INR r	0r4	2.5	(r)←(r)+1 The content of register r is incremented by one. All the condition flip-flops except carry are affected by the result.
DCR r	0r5	2.5	(r)←(r)-1 The content of register r is decremented by one. All of the condition flip-flops except carry are affected by the result.
INR M	064	5.0	[M]←[M]+1. The content of memory designated by registers H and L is incremented by one. All of the condition flip-flops except carry are affected by the result.
DCR M	065	5.0	[M]←[M]-1. The content of memory designated by registers H and L is decremented by one. All of the condition flip-flops except carry are affected by the result.
ADD r	20r	2.0	(A)←(A)+(r) Add the content of register r to the content of register A and place the result into register A. (All flags affected.)

Mnemonic	Op Code (octal)	Execution Time (μs)	Description of Operation
ADC r	21r	2.0	(A)←(A)+(r)+(carry) Add the content of register r and the contents of the carry flip-flop to the content of the A register and place the result into Register A. (All flags affected.)
SUB r	22r	2.0	(A)←(A)-(r) Subtract the content of register r from the content of register A and place the result into register A. Two's complement subtraction is used. (All flags affected.)
SBB r	23r	2.0	(A)←(A)-(r)-(borrow) Subtract the content of register r and the content of the carry flip-flop from the content of register A and place the result into register A. (All flags affected.)
ANA r	24r	2.0	(A)←(A)·(r) Place the logical product of the register A and register r into register A. (Resets carry.)
XRA r	25r	2.0	(A)←(A)⊕(r) Place the "exclusive-or" of the content of register A and register r into register A. (Resets carry.)
ORA r	26r	2.0	(A)←(A)∨(r) Place the "inclusive-or" of the content of register A and register r into register A. (Resets carry.)
CMP r	27r	2.0	(A)-(r) Compare the content of register A with the content of register r. The content of register A remains unchanged. The flag flip-flops are set by the result of the subtraction. Equality (A=r) is indicated by the zero flip-flop set to "1." Less than (A<r) is indicated by the carry flip-flop, set to "1."
ADD M	206	3.5	(A)←(A)+(M) ADD
ADC M	216	3.5	(A)←(A)+(M)+(carry) ADD with carry
SUB M	226	3.5	(A)←(A)-(M) SUBTRACT
SBB M	236	3.5	(A)←(A)-(M)-(borrow) SUBTRACT with borrow
ANA M	246	3.5	(A)←(A)·(M) Logical AND
XRA M	256	3.5	(A)←(A)⊕(M) Exclusive OR
ORA M	266	3.5	(A)←(A)∨(M) Inclusive OR

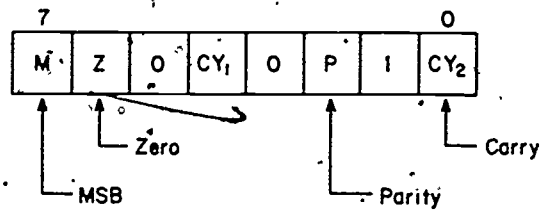
Mnemonic	Op Code (octal)	Execution Time (μs)	Description of Operation
CMP M	276	3.5	(A)-(M) COMPARE
ADI $<B_2>$	306	3.5	$(A) \leftarrow (A)+<B_2>$ ADD
ACI $<B_2>$	316	3.5	$(A) \leftarrow (A)+<B_2>+$ (carry) ADD with carry
SUI $<B_2>$	326	3.5	$(A) \leftarrow (A)-<B_2>$ SUBTRACT
SBI $<B_2>$	336	3.5	$(A) \leftarrow (A)-<B_2>-$(borrow) SUBTRACT with borrow
ANI $<B_2>$	346	3.5	$(A) \leftarrow (A) \cdot <B_2>$ Logical AND
XRI $<B_2>$	357	3.5	$(A) \leftarrow (A) \oplus <B_2>$ Exclusive OR
ORI $<B_2>$	366	3.5	$(A) \leftarrow (A) \vee <B_2>$ Inclusive OR
CPI $<B_2>$	376	3.5	$(A)-<B_2>$ COMPARE
RLC	007	2.0	$A_{i+1} \leftarrow A_i, A_0 \leftarrow A_7$, (carry)$\leftarrow A_7$ Rotate the content of register A left one bit. Rotate A_7 into A_0 and into the carry flip-flop.
RRC	017	2.0	$A_i \leftarrow A_{i+1}, A_7 \leftarrow A_0$, (carry)$\leftarrow A_0$ Rotate the content of register A right one bit. Rotate A_0 into A_7 and into the carry flip-flop.
RAL	027	2.0	$A_{i+1} \leftarrow A_i, A_0 \leftarrow$ (carry), (carry)$\leftarrow A_7$ Rotate the content of Register A left one bit. Rotate the content of the carry flip-flop into A_0. Rotate A_7 into the carry flip-flop.
RAR	037	2.0	$A_i \leftarrow A_{i+1}, A_7 \leftarrow$ (carry), (carry)$\leftarrow A_0$ Rotate the content of register A right one bit. Rotate the content of the carry flip-flop into A_7. Rotate A_0 into the carry flip-flop.
JMP $<B_3>$ $<B_2>$	303	5.0	$(PC) \leftarrow <B_3><B_2>$ Jump unconditionally to the instruction located in memory location addressed by byte two and byte three.

Mnemonic	Op Code (octal)	Execution Time (μs)	Description of Operation
JC <B_2> <B_3>	332	5.0	If (Carry)=1 (PC)←<B_3><B_2> Otherwise (PC)=(PC)+3
JNC <B_2> <B_3>	322	5.0	If (Carry)=0 (PC)←<B_3><B_2> Otherwise (PC)=(PC)+3
JZ <B_2> <B_3>	312	5.0	If (Zero)=1 (PC)←<B_3><B_2> Otherwise (PC)=(PC)+3
JNZ <B_2> <B_3>	302	5.0	If (Zero)=0 (PC)←<B_3><B_2> Otherwise (PC)=(PC)+3
JP <B_2> <B_3>	362	5.0	If (Sign)=0 (PC)←<B_3><B_2> Otherwise (PC)=(PC)+3
JM <B_2> <B_3>	372	5.0	If (Sign)=1 (PC)←<B_3><B_2> Otherwise (PC)=(PC)+3
JPE <B_2> <B_3>	352	5.0	If (Parity)=1 (PC)←<B_3><B_2> Otherwise (PC)=(PC)+3
JPO <B_2> <B_3>	342	5.0	If (Parity)=0 (PC)←<B_3><B_2> Otherwise (PC)=(PC)+3
CALL <B_2> <B_2>	315	8.5	[SP-1][SP-2]←(PC), (SP)=(SP)-2 (PC)←<B_3><B_2> Transfer the content of PC to the pushdown stack in memory addressed by the register SP. The content of SP is decremented by two. Jump unconditionally to the instruction located in memory location addressed by byte two and byte three of the instruction.
CC <B_2> <B_3>	334	5.5/8.5	If (carry)=1 [SP-1][SP-2]←PC, (SP)=(SP)-2, (PC)←<B_3><B_2>; otherwise (PC)=(PC)+3
CNC <B_2> <B_3>	324	5.5/8.5	If (carry)=0 [SP-1][SP-2]←PC, (SP)=(SP)-2, (PC)←<B_3><B_2>; otherwise (PC)=(PC)+3

Mnemonic	Op Code (octal)	Execution Time (μs)	Description of Operation
CZ <B_2> <B_3>	314	5.5/8.5	If (zero)=1[SP-1][SP-2]←PC, (SP)=(SP)-2,(PC)←<B_3><B_2>; otherwise (PC)=(PC)+3
CNZ <B_2> <B_3>	304	5.5/8.5	If (zero)=0[SP-1][SP-2]←PC, (SP)=(SP)-2,(PC)←<B_3><B_2>; otherwise (PC)=(PC)+3
CP <B_2> <B_3>	364	5.5/8.5	If (sign)=0[SP-1][SP-2]←PC, (SP)=(SP)-2,(PC)←<B_3><B_2>; otherwise (PC)=(PC)+3
CM <B_2> <B_3>	374	5.5/8.5	If (sign)=1[SP-1][SP-2]←PC, (SP)=(SP)-2,(PC)←<B_3><B_2>; otherwise (PC)=(PC)+3
CPE <B_2> <B_3>	354	5.5/8.5	If (parity)=1[SP-1][SP-2]←PC, (SP)=(SP)-2,(PC)←<B_3><B_2>; otherwise (PC)=(PC)+3
CPO <B_2> <B_3>	344	5.5/8.5	If (parity)=0[SP-1][SP-2]←PC, (SP)=(SP)-2,(PC)←<B_3><B_2>; otherwise (PC)=(PC)+3
RET	311	5.0	(PC)←[SP][SP+1](SP)=(SP)+2. Return to the instruction in the memory location addressed by the last values shifted into the pushdown stack addressed by SP.
RC	330	2.5/5.5	If (carry)=1(PC)←[SP],[SP+1], (SP)=(SP)+2; otherwise (PC)=(PC)+1
RNC	320	2.5/5.5	If (carry)=0(PC)←[SP],[SP+1], (SP)=(SP)+2; otherwise (PC)=(PC)+1
RZ	310	2.5/5.5	If (zero)=1(PC)←[SP],[SP+1], (SP)=(SP)+2; otherwise (PC)=(PC)+1
RNZ	300	2.5/5.5	If (zero)=0(PC)←[SP],[SP+1], (SP)=(SP)+2; otherwise (PC)=(PC)+1
RP	360	2.5/5.5	If (sign)=0(PC)←[SP],[SP+1], (SP)=(SP)+2; otherwise (PC)=(PC)+1

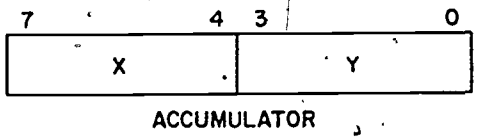
Mnemonic	Op Code (octal)	Execution Time (μs)	Description of Operation
RM	370	2.5/5.5	If (sign)=1 (PC)←[SP], [SP+1], (SP)=(SP)+2; otherwise (PC)=(PC)+1
RPE	350	2.5/5.5	If (parity)=1 (PC)←[SP], [SP+1], (SP)=(SP)+2; otherwise (PC)=(PC)+1
RPO	340	2.5/5.5	If (parity)=0 (PC)←[SP], [SP+1], (SP)=(SP)+2; otherwise (PC)=(PC)+1
RST	3A7	5.5	[SP-1] [SP-2]←(PC), (SP)=(SP)-2 (PC)←(0000A0)
IN <B_2>	333	5.0	(A)←INPUT data Input a byte of data to A from the device specified by (B_2). See Input-Output sections for IO address assignments.
OUT <B_2>	323	5.0	OUTPUT←(A) Output the contents of the accumulator to the device specified by (B_2). See Input-Output section for IO address assignments.
LXI B <B_2> <B_3>	001	5.0	(C)←<B_2>; (B)←<B_3> Load byte two of the instruction into C. Load byte three of the instruction into B.
LXI D <B_2> <B_3>	021	5.0	(E)←<B_2>, (D)←<B_3> Load byte two of the instruction into E. Load byte three of the instruction into D.
LXI H <B_2> <B_3>	041	5.0	(L)←<B_2>, (H)←<B_3> Load byte two of the instruction into L. Load byte three of the instruction into H.
LXI SP <B_2> <B_3>	061	5.0	$(SP)_L$←<B_2>, $(SP)_H$←<B_3> Load byte two of the instruction into the lower order 8-bit of the stack pointer and byte three into the higher order 8-bit of the stack pointer.

Mnemonic	Op Code (octal)	Execution Time (s)	Description of Operation
PUSH PSW	356	5.5	[SP-1]←(A),[SP-2]←(F),(SP)=(SP)-2 Save the contents of A and F (5-flags) into the pushdown stack addressed by the SP register. The content of SP is decremented by two. The flag word will appear as follows:
PUSH B	305	5.5	[SP-1]←(B) [SP-2]←(C),(SP)=(SP)-2
PUSH D	325	5.5	[SP-1]←(D) [SP-2]←(E),(SP)=(SP)-2
PUSH H	345	5.5	[SP-1]←(H) [SP-2]←(L),(SP)=(SP)-2
POP PSW	361		(F)←[SP],(A)←[SP+1],(SP)=(SP)+2 Restore the last values in the push-down stack addressed by SP into A and F. The content of SP is incremented by two.
POP B	301	5.0	(C)←[SP],(B)←[SP+1],(SP)=(SP)+2
POP D	321	5.0	(E)←[SP],(D)←(SP+1],(SP)=(SP)+2
POP H	341	5.0	(L)←[SP],(H)←[SP+1],(SP)=(SP)+2
STA <B_2> <B_3>	032	6.5	[<B_3><B_2>]←(A) Store the accumulator content into the memory location addressed by byte two and byte three of the instruction.
LDA <B_2> <B_3>	072	6.5	(A)←[<B_3><B_2>] Load the accumulator with the content of the memory location addressed by byte two and byte three of the instruction.
XCHG	353	2.0	(H)↔(D) (E)↔(L) Exchange the contents of registers H and L and registers D and E.



Mnemonic	Op Code (octal)	Execution Time (μs)	Description of Operation
XTHL	343	9.0	(L)↔[SP], (H)↔[SP+1] Exchange the contents of registers H, L and the last values in the pushdown stack addressed by registers SP. The SP register itself is not changes. (S)=(SP)
SPHL	374	2.5	(SP)←(H)(L) Transfer the contents of registers H and L into register SP.
PCHL	351	2.5	(PC)←(H)(L) JUMP INDIRECT
DAD B	011	5.0	(H)(L)←(H)(L)+(B)(C)
DAD D	031	5.0	(H)(L)←(H)(L)+(D)(E)
DAD H	051	5.0	(H)(L)←(H)(L)+(H)(L) (double precision shift left H and L)
DAD SP	071	5.0	(H)(L)←(H)(L)+(SP) Add the content of register SP to the content of registers H and L and place the result into registers H and L. If the overflow is generated, the carry flip-flop is reset. The other condition flip-flops are not affected. This is useful for addressing data in the stack.
STAX B	002	3.5	[(B)(C)]←(A) Store the accumulator content in the memory location addressed by the content of registers B and C.
STAX D	022	3.5	[(D)(E)]←(A) Store the accumulator content into the memory location addressed by the content of register D and E.
LDAX B	012	3.5	(A)←[(B)(C)] Load the accumulator with the content of the memory location addressed by the content of registers B and C.
LDAX D	032	3.5	(A)←[(D)(E)] Load the accumulator with the content of memory location addressed by the content of register D and E.

Mnemonic	Op Code (octal)	Execution Time (μs)	Description of Operation
INX B	003	2.5	(B)(C)←(B)(C)+1 The content of register pair B and C is incremented by one. All of the condition flip-flops are not affected.
INX D	023	2.5	(D)(E)←(D)(E)+1
INX H	043	2.5	(H)(L)←(H)(L)+1 The content of register H and L is incremented by one. All of the condition flip-flops are not affected.
INX SP	063	2.5	(SP)←(SP)+1
DCX B	013	2.5	(B)(C)←(B)(C)-1
DCX H	053	2.5	(H)(L)←(H)(L)-1
DCX D	033	2.5	(D)(E)←(D)(E)-1
DCX SP	073	2.5	(SP)←(SP)-1
CMA	057	2.0	(A)←($\overline{A}$) The content of accumulator is complemented. The condition flip-flops are not affected.
STC	067	2.0	(Carry)←1 Set the carry flip-flop to 1. The other condition flip-flops are not affected.
CMC	077	2.0	(carry)←($\overline{carry}$) The content of carry is complemented. The other condition flip-flops are not affected.

DAA	047	2.0	Decimal Adjust Accumulator
			The 8-bit value in the accumulator containing the result from an arithmetic operation on decimal operands is adjusted to contain two valid BCD digits by adding a value according to the following rules:
			 
			
			ACCUMULATOR
			If (Y≥10) or (carry from bit 3) then Y=Y+6 with carry to X digit.
			If (X≥10) or (carry from bit 7) or [(Y≥10) and (X=9)] then X=X+6 (which sets the carry flip-flop).

Mnemonic	Op Code (octal)	Execution Time (μs)	Description of Operation
DAA (cont.)			Two carry flip-flops are used for this instruction. CY_1 represents the carry from bit 3 (the fourth bit) and is accessible as a fifth flag. CY_2 is the carry from bit 7 and is the usual carry bit. All condition flip-flops are affected by this instruction.
SHLD <B_2> <B_3>	042	8.0	$[<B_3><B_2>]\leftarrow(L),[<B_3><B_2>+1]\leftarrow(H)$ Store the contents of registers H and L into the memory location addressed by the byte two and byte three of the instructions.
LHLD <B_2> <B_3>	052	8.0	$(L)\leftarrow[<B_3><B_2>],(H)\leftarrow[<B_3><B_2>+1]$ Load the registers H and L with the contents of the memory location addressed by byte two and byte three of the instruction.
EI	373	2.0	Interrupt System Enable
DI	363	2.0	Interrupt System Disable The terminal interrupt system can be enabled or disabled using the above instructions. An interrupt will be accepted only if the interrupt system is enabled. Upon receipt of an interrupt, the interrupt system will be automatically disabled. It must be re-enabled by the program. During execution of Enable or Disable Interrupt instructions, an interrupt will not be accepted.
NOP	000	2.0	No operation

5. Input-Output

Input and output operations are performed by the IN and OUT instructions and involve the transmission of 8-bit data words between the accumulator (A) register and devices external to the terminal processor. In some cases the OUT instruction is used to set control flags in the Panel Interface Unit (PIU) and no data is actually transferred. In these cases the contents of A are immaterial.

The format of the I/O address is shown in Figure 5.0.

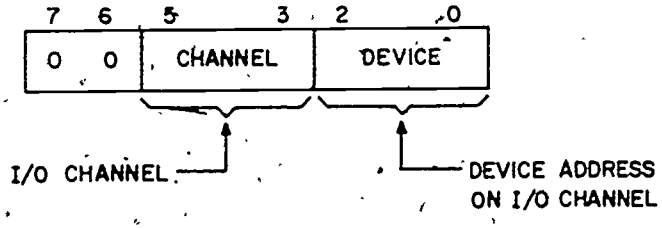


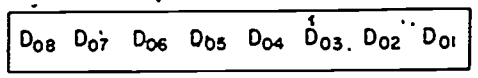
Figure 5.0. I/O Address Format

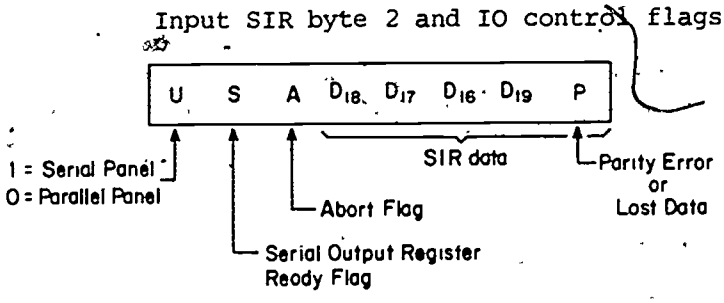
I/O Channels 0-3 are used internally; and Channels 4-7, externally.

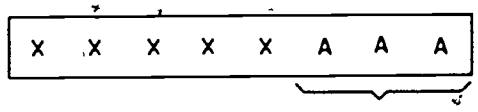
The input and output device address assignments and functions are tabulated on the following pages.

Input

Input Address (octal)	Mnemonic	Function
00	SIRØ	Input SIR byte 0
01	SIR1	Input SIR byte 1
02	SIR2	Input SIR byte 2 and IO control flags
03	INTVECT	Input interrupt vector
04	KST	Input keyset word
05-17	unused	
20	XL	Input lower 8 bits of x register
21	XU	Input most significant bit (x_8) of x register to A_0
22	YL	Input lower 8 bits of y register

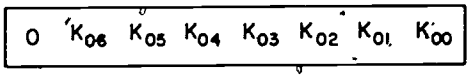




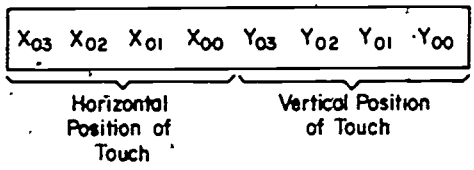


Interrupt Device Address as Follows:

000	SIR
001	KST
010	TP
011	SOR
100	EXT ϕ
101	EXT 1
110	EXT 2
111	EXT 3



Input Address (octal)	Mnemonic	Function
23	YU	Input most significant bit (y_8) of y register to A_0.
24-37	unused	
40	TP	Input touch panel word
42-44	unused	
45	EXT 0	Input word from external device 0
46-77	unused	

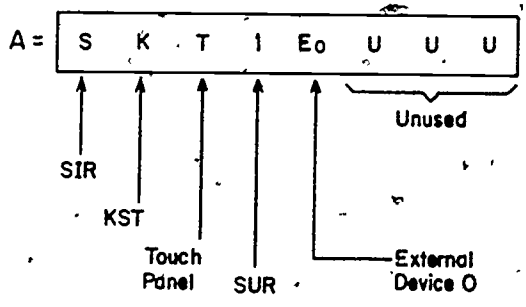


Output

Output Address (octal)	Mnemonic	Function
00	SOR	Load the lower 8 bits of the SOR with the contents of A and transmit the entire contents (12 bits) of the SOR to the central computer. The upper 4 bits of the SOR are assumed to have been previously loaded by the OUT SORI instruction.
01	SORI	Set the SOR message start bit (bit 11) to "1" and load bits 8-10 of the SOR with the lower 3 bits of A.
02	RESET	Reset the message start flag in the demodulator section of the modem. The contents of A are unused.
03-05	unused	
06	EXTRDY	Generate EXTERNAL DATA READY signal for serial output channel data. The contents of A are unused.
07	IMASK	Load the interrupt mask register with the contents of A.
10	XLONG	Set the long vector in the PIU to x. The contents of A are unused. See OUT CLOCKL instruction for use of this flag.
11	YLONG	Set the long vector flag in the PIU to y. The contents of A are unused. See OUT CLOCKL instruction for use of this flag.

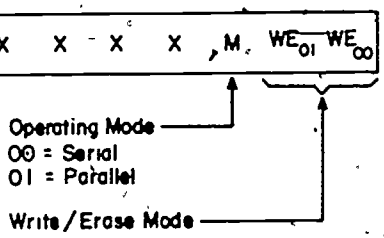
A = | D_{07} | D_{06} | D_{05} | D_{04} | D_{03} | D_{02} | D_{01} | D_{00} |

A = | X | X | X | X | X | D_{10} | D_{09} | D_{08} |



Output Address (octal)	Mnemonic	Function
12	SETXR	Set the x direction flag in the PIU. The x register will be decremented by all subsequent clock x signals. The contents of A are unused.
13	SETXF	Reset (CLEAR) the x direction flag in the PIU. The x register will be incremented by all subsequent clock x signals. The contents of A are unused.
14	SETYR	Set the y direction flag in the PIU. The y register will be decremented by all subsequent clock y signals. The contents of A are unused.
15	SETYF	Reset (CLEAR) the y direction flag in the PIU. The y register will be incremented by all subsequent clock y signals. The contents of A are unused.
16	SETABT	Set the ABORT flag. This instruction places the terminal in the ABORT mode. The contents of A are unused.
17	CLRABT	Reset (CLEAR) the ABORT flag. This instruction places the terminal in the normal operating mode. The contents of A are unused.
20	XL	Load the lower 8 bits of the x register with the contents of A.
21	XU	Load the most significant bit (x_8) of the x register with bit A_0 of the accumulator. The other bits of A are unused.
22	YL	Load the lower 8 bits of the y register.
23	YU	Load the most significant bit (y_8) of the y register with bit A_0 of the accumulator. The other bits of A are unused.
24	PDL	Load the lower 8 bits of the panel parallel data register with the contents of A.
25	PDU	Load the upper 8 bits of the panel parallel data register with the contents of A and write (or erase) the contents of the parallel data register (16 bits) on the panel.

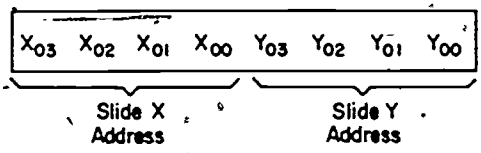
Output Address (octal)	Mnemonic	Function
26	PDM	Load the PIU mode register with the lower 3 bits of A.



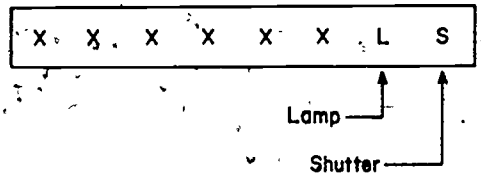
27	PDLU	Load both PDL and PDU with the contents of A and write (or erase) the contents (16 bits) on the panel.
30	CLOCKX	Clock the x register. The x direction flag specifies the direction; forward if reset (0), reverse if set (1). The contents of A are unused.
31	CLOCKY	Clock the y register. The y direction flag specifies the direction; forward if reset (0), reverse if set (1). The contents of A are unused.
32	CLOCKXY	Clock both the x and y registers and write (or erase) the resulting address on the panel. The x and y direction flags specify direction of change. The contents of A are unused.
33	CLOCKL	Clock the long vector, x or y, (as specified by the long vector flag) in the PIU if A_0=0; clock both the x and y registers if A_0=1. The resulting address is then written (or erased) on the panel. The direction flags specify direction of change. The other bits of A are unused.
34	HCHAR	Clock the y register and write (or erase) the resulting address on the panel. The contents of A_0 and the WE bits in the Panel Mode register specify the operation performed as follows:

A_0	WE_1	WE_0	FUNCTION
0	0	0	write
0	0	1	erase
0	1	0	nop
0	1	1	nop
1	0	0	erase
1	0	1	write
1	1	0	erase
1	1	1	write

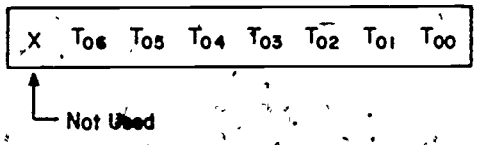
Output Address (octal)	Mnemonic	Function
35	VCHAR	Clock the x register and write (or erase) the resulting address on the panel as shown in the table for OUT HCHAR instruction.
36	WE	Write if WE_0=1 or erase if WE_0=0 the address specified by the contents of the x and y registers. The contents of A are unused.
37	SCREEN	Erase the entire panel. The contents of A are unused.
40	SLIDEL	Load the lower 8 bits of the slide projector register with the contents of A.



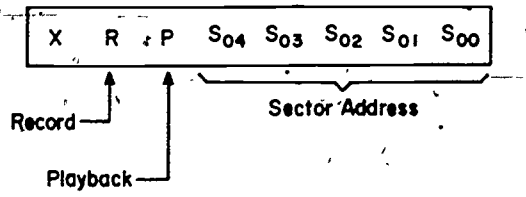
41	SLIDEU	Load the upper 2 bits of the slide projector with the lower 2 bits of A.



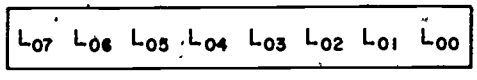
42	AUDIOT	Load the audio unit Track Address register with the contents of A.



43	AUDIOS	Load the audio unit Sector Address register and the Record/Playback flags with the contents of A.



44	AUDIOL	Load the audio unit Message Length Register with the contents of A and play the message addressed by the Track and Sector registers.

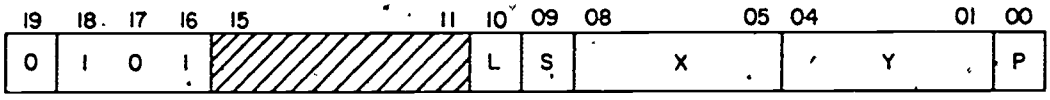


Output Address (octal)	Mnemonic	Function
45	EXT0	Load external device 0 with the contents of A.
46	EXT1	Load external device 1 with the contents of A.
47	EXT2	Load external device 2 with the contents of A.
50-77	unused	

APPENDIX

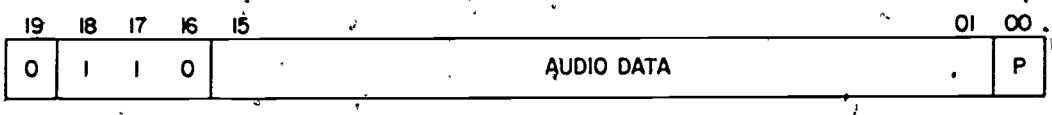
In the present PLATO terminal, the D = 101, 110, 111 control words have the formats shown below.

D = 101 (SSL) Load Slide (223μs)

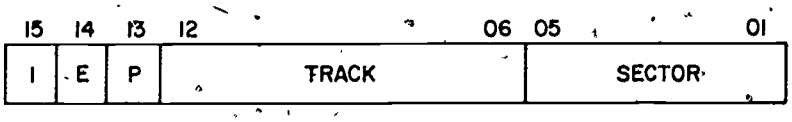


This instruction is used to operate the slide projector. Bits 01 - 08 select one of 256 slides for display on the plasma panel. Bit 09 controls the projector shutter. For normal operation this bit is always "0". However, if this bit is a "1", the shutter will be closed and remain closed until receipt of a load slide command with bit 09 = "0". Bit 10 controls the projector lamp. The lamp will be turned on if bit 10 is a "1" and off if bit 10 is a "0".

D = 110 (AUD) Load Audio (223μs)



This instruction is used to control the audio response unit. The audio response unit requires two of these instruction per audio operation. The formats of each of these instruction are described below.

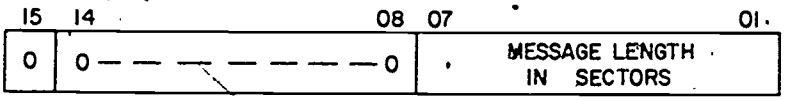


First audio instruction

Bits 01 - 12 — Specify the message starting address; bits 01 - 05 specify one of 32 sectors, and bits 06 - 12, one of 128 tracks.

Bits 13 - 14 — Specify playback or erase as follows:
- 00 - do nothing
- 01 - play message
- 10 - do nothing
- 11 - record message

Bit 15 — Always "1". Identifies first of two audio instructions.



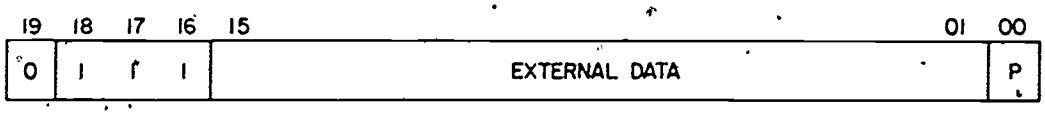
Second audio instruction

Bits 01 - 07 — Specify length of message in terms of sectors. One sector equals ∿ 1/3 seconds.

Bits 08 - 14 — Unused

Bit 15 — Always "0" - identifies second audio instruction.

D = 111 (EXT) Load External (275μs)



This instruction transfers bits 01 - 15 to any equipment attached to the external output channel of the terminal.